Lab 1

Lab 1

For this lab, you will create a `Student` class that has the following `private` class attributes:

- `string name` - name of the student
- `int grade` - student's grade level
- `int score` - test score of the student

First, let's put together all of the class attributes.

```
class Student {
  private:
    string name;
    int grade;
    int score;
};
```

Next, let's define the constructor `Student` with two parameters `string n` for the student's name and `int g` for the student's grade.

```
class Student {
  public:
    Student(string n, int g) {
      name = n;
      grade = g;
    }

  private:
    string name;
    int grade;
    int score;
};
```

Now, you want to define a class function called `StudentStatus` that takes a student's `int score` as a parameter and checks whether the score is a passing score or not. A passing score is `65` or higher. If the score is less than `65`, then the student did not pass their grade and will remain in the same grade they are now. Otherwise, if the student has a `score` of `65` or higher, then they have passed and will move on to the next grade and their `grade`

attribute will increase by 1. The function should also output a message providing some context regarding whether the student has been promoted to the next grade or not.

```
//add class definitions below this line

class Student {
  public:
    Student(string n, int g) {
      name = n;
      grade = g;
    }
    void StudentStatus(int s) {
      if (s < 65) {
        score = s;
        cout << name << " has not graduated and will remain in
        grade ";
        cout << grade << "." << endl;
      }
      else {
        score = s;
        cout << name << " has graduated and will be promoted to
        grade ";
        cout << grade + 1 << "." << endl;
      }
    }

  private:
    string name;
    int grade;
    int score;
};

//add class definitions above this line
```

In `main`, try a few test cases to see if `StudentStatus` updates `alice`'s grade level correctly.

```
  //add code below this line

  Student alice("Alice", 4);
  alice.StudentStatus(60);
  alice.StudentStatus(90);

  //add code above this line
```

Since `Alice` received a score of `65` the first time, they were not promoted to the next grade. However, once they got a score of `90` the second time around, they were promoted to the next grade.

Lab 2

Lab 2

It is important to understand why class attributes are labeled as `private`. This provides a level of protection for your code since it does not allow the user to interact with the class attributes directly.

```
//add class definitions below this line

class Student {
  public:
    Student() {
      name;
      grade;
    }

  public:
    string name = "Alice";
    int grade = 4;
    int score = 65;
};

//add class definitions above this line
```

Because the code above has `public` class attributes, the following code in `main` can change those attributes' values directly.

```
  //add code below this line

  Student steve;
  steve.name = "Steve";
  cout << steve.name << endl;

  //add code above this line
```

However, if you change the class attributes from `public` to `private`, the code in `main` will no longer work.

```
//add class definitions below this line

class Student {
  public:
    Student() {
      name;
      grade;
    }

  private:
    string name = "Alice";
    int grade = 4;
    int score = 65;
};

//add class definitions above this line
```

```
  //add code below this line

  Student steve;
  steve.name = "Steve";
  cout << steve.name << endl;

  //add code above this line
```

This is why understanding how class functions work is important. Class functions serve as the intermediate step between the objects and the class attributes. They are the ones interacting with the class attributes instead of the user.

```
//add class definitions below this line

class Student {
  public:
    Student() {
      name;
      grade;
    }
    void ChangeName(string n) {
      name = n;
    }
    string ReturnName() {
      return name;
    }


  private:
    string name = "Alice";
    int grade = 4;
    int score = 65;
};

//add class definitions above this line
```

```
  //add code below this line

  Student steve;
  steve.ChangeName("Steve");
  cout << steve.ReturnName() << endl;

  //add code above this line
```

Although using class functions may result in longer code, it prevents the user from seeing and interacting with the class attributes directly. This is why using class attributes is a best practice.

Lab Challenge

Copy and paste the `Zoo` class below into the text editor.

```
//add class definitions below this line

class Zoo {
  public:
    Zoo(int bc, int p, int r, int b) {
      big_cats = bc;
      primates = p;
      reptiles = r;
      birds = b;
    }

  private:
    int big_cats; //for "big cats"
    int primates; //for "primates"
    int reptiles; //for reptiles
    int birds; //for birds
};

//add class definitions above this line
```

Your task is to add the following class functions to the class:
* `TotalAnimals` - returns the total number of animals
* `TotalMammals` - returns the number of mammals (which includes `big_cats` and `primates`)
* `MostAnimals` - returns the name of the animal with the most count assuming no two animals have the same count

DO NOT CHANGE the existing code in `main` or you will not pass the test:

```
  Zoo my_zoo(10, 30, 90, 120);
  cout << my_zoo.TotalAnimals() << endl;
  cout << my_zoo.TotalMammals() << endl;
  cout << my_zoo.MostAnimals() << endl;
  Zoo my_zoo2(123, 45, 67, 89);
  cout << my_zoo2.TotalAnimals() << endl;
  cout << my_zoo2.TotalMammals() << endl;
  cout << my_zoo2.MostAnimals() << endl;
```

Expected Result:

```
250
40
birds
324
168
big cats
```